A Methodology for Fitting and Validating Metamodels in Simulation

Kleijnen, J.P.C.; Sargent, R.

Publication date:
1997

U:\BIKASECR\KLEIJNEN\SARGCENT.WPD

A Methodology for Fitting and Validating Metamodels in Simulation

by

Jack P.C. Kleijnen [a] and **Robert G. Sargent** [b]

[a] Department of Information Systems/Center for Economic Research (CentER)

Tilburg University, 5000 LE Tilburg, Netherlands

E-mail: kleijnen@kub.nl

Fax: +3113-4663377

Web: http://cwis.kub.nl/~few5/center/staff/kleijnen/cv2.htm

[b] Simulation Research Group

Department of Electrical Engineering and Computer Science

439 Link Hall, Syracuse University, Syracuse NY 13244, USA

E-mail: rsargent@syr.edu

Fax: 315 443 4936

Acknowledgment:

CentER provided financial support enabling R.G. Sargent to visit Tilburg University during one month for research on the topic of this paper.

Gűl Gűrkan of Tilburg University provided useful comments on a previous draft.

A methodology for fitting and validating metamodels in simulation

Jack P.C. Kleijnen [a] and Robert G. Sargent [b c]

[a] Department of Information Systems/Center for Economic Research (CentER), Tilburg University, 5000 LE Tilburg, Netherlands, E-mail: kleijnen@kub.nl, Fax: +3113-4663377, Web: http://cwis.kub.nl/~few5/center/staff/kleijnen/cv2.htm

[b] Simulation Research Group, Department of Electrical Engineering and Computer Science, 439 Link Hall, Syracuse University, Syracuse NY, 13244, USA, E-mail: rsargent@syr.edu, Fax: 315 443 4936

[c] Acknowledgment: CentER provided financial support enabling R.G. Sargent to visit Tilburg University during one month for research on the topic of this paper

Abstract

This expository paper discusses the relationships among metamodels, simulation models, and problem entities. A metamodel or response surface is an approximation of the input/output function implied by the underlying simulation model. There are several types of metamodel: linear regression, splines, neural networks, etc. This paper distinguishes between fitting and validating a metamodel. Metamodels may have different goals: (i) understanding, (ii) prediction, (iii) optimization, and (iv) verification and validation. For this metamodeling, a process with thirteen steps is proposed. Classic design of experiments (DOE) is summarized, including standard measures of fit such as the R-square coefficient and cross-validation measures. This DOE is extended to sequential or stagewise DOE. Several validation criteria, measures, and estimators are discussed. Metamodels in general are covered, along with a procedure for developing linear regression (including polynomial) metamodels.

Keywords

Simulation, Approximation, Response surface, Modelling, Regression

1. Introduction

There is a growing interest in metamodels that are developed from simulation models for problem entities. Yet -to the best of our knowledge- there are no publications that give a

complete methodology for fitting and validating such metamodels; we do give a comprehensive methodology in §2 and Figure 3.

Figure 1 shows the relationships we see among these three concepts; note that WRT stands for 'with respect to'. A problem entity is some system (real or proposed), idea, situation, policy, or phenomena that is being modeled. A simulation model is a causal model of some problem entity; this model may be deterministic or stochastic. A metamodel, as the term is used here, is an approximation of the input/output (I/O) transformation that is implied by the simulation model; the resulting black-box model is also known as a response surface. There are different types of metamodels; for example, polynomial regression models (which are a type of linear regression), splines (which partition the domain of applicability into subdomains and fit simple regression models to each of the subdomains), and neural networks (a type of non-linear regression); see Barton (1993, 1994), Friedman (1996), Huber et al. (1996), Kleijnen (1998), Pierreval (1996), Yu and Popplewell (1994).

INSERT Figure 1: Metamodel, simulation model, and problem entity

Although metamodels have been quite frequently applied in the simulation practice and studied in the simulation literature, this has been done in an *ad hoc* way. This paper offers a methodology for developing metamodels; this methodology distinguishes between fitting and validating a metamodel.

We use the term *validation* as it is commonly used in the simulation literature (e.g., Sargent 1996 and Schlesinger et al. 1979); that is, model validation is the 'substantiation that a model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model'. Figure 1 contains a view of how validation relates to problem entity, simulation model, and metamodel. The validation of a metamodel relates to both the simulation model and the problem entity. To validate a metamodel, we must know the amount of accuracy required of the metamodel; this amount depends on the use of the metamodel. The validation of *simulation* models will not be discussed in this paper, except when discussing the use of a metamodel to aid in this validation. (There is considerable literature on validation of simulation models; see, e.g., Kleijnen 1995b, Sargent 1996, and http://manta.cs.vt.edu/biblio/.) In the remainder of this paper we shall assume that the simulation model under consideration is valid, unless stated otherwise.

Fitting, as we use the term in this paper, is the process of applying mathematical and statistical techniques to a set of I/O data from a simulation model, in order to (i) estimate the metamodel's parameter values, and (ii) evaluate the estimated parameter values with respect to the data set, using quantitative criteria. Prior to fitting a metamodel, we must specify the type and form (or class) of the metamodel. Validation requires knowledge about the problem entity (domain knowledge) and the specified amount of accuracy required of the metamodel. Fitting, on the other hand, is concerned only with determining a 'good' set of parameter values from the data set, and is not concerned with whether the metamodel adequately describes the problem entity and the simulation model for the goal of the metamodel.

Any model should be developed for a specific *goal*. For metamodels we identify four general goals: (i) obtaining an understanding of the problem entity, (ii) predicting values of the output or response variable, (iii) performing optimization, and (iv) aiding in the verification and validation (V & V) of a simulation model. The goal of the metamodel needs to be used when specifying the type and form of metamodel to be fitted and when determining the validity of the metamodel.

Current practice in metamodeling uses metamodels that have a single output. In practice, however, the study of a problem entity often distinguishes several outputs of interest. Hence, the corresponding simulation model also has more than one response variable. In such a case a separate metamodel is developed for each of these outputs. Indeed we develop a methodology for metamodels with a single output. This methodology can be applied to each of the single-response metamodels. Figure 2 displays a single output for problem entity, simulation models, and metamodel respectively.

INSERT Figure 2: I/O data of problem entity, simulation model, and metamodel

The remainder of this paper is organized as follows. Section 2 presents the methodology for developing metamodels, i.e., the metamodeling process. Section 3 discusses some issues relevant to developing metamodels. Section 4 focusses on the development of linear-regression metamodels. Section 5 contains some research issues. Section 6 provides a summary.

2. The Metamodeling Process

We suggest the following thirteen steps for developing a metamodel.

1. Determine the goal of the metamodel

The goals of the metamodel must be determined, and the metamodel must be developed for those goals. Let us consider the four goals mentioned above.

(a) Problem entity understanding

A goal may be understanding the internal behavior of a problem entity (e.g., determining the 'bottlenecks' in a system), understanding the behavior of the output, performing sensitivity analysis (i.e., determining how sensitive the output is to changes to the inputs), and conducting what-if studies.

(b) Prediction

The metamodel may replace the simulation model to obtain a set of values of the output for a specific set of values of the inputs. The metamodel is then used routinely instead of the simulation, since the metamodel is usually quicker and easier to use than the simulation model.

(c) Optimization

The metamodel may be used to determine the set of values of the problem entity inputs that optimizes a specific objective function. The objective function may be either one of the problem entity's outputs or a function that contains one or more of these outputs.

(d) Aid in V & V of the simulation model

The metamodel may be developed to assist in the V & V of the simulation model.

2. Identify the inputs and their characteristics

We distinguish between input variables (briefly: inputs) and parameters (see Zeigler 1976). Inputs are directly observable (for example, number of servers), whereas parameters require statistical inference (for example, service rate λ). (The simulation computer program may treat these inputs and parameters as 'inputs'.) For each input we should determine whether the input is deterministic or random; also we must select the type of measurement scale (see §3.2). The metamodel has 'independent variables' x_g, which are functions of the simulation inputs and parameters; for example, $x_1 = \lambda^2$; see Figure 2. The independent variables of the metamodel should be identified. The simulation inputs and parameters are called factors in the design of experiments (DOE).

Note: Even a random simulation model has deterministic inputs and parameters; the only random element is s_0, the seed of its pseudorandom number generator, but even this seed may be fixed.

3. Specify the domain of applicability (experimental region)

We should determine the combination of values of the independent variables for which the metamodel is to be used, i.e., the domain of applicability for which the metamodel is to be valid. (To obtain a valid simulation model, its factor values also need to be restricted to a certain domain; also see the 'experimental frame' concept in Zeigler 1976.)

4. Identify the output variable and its characteristics

The output should first be identified. Next (analogous to the inputs), this output must be classified as random or deterministic, and a measurement scale must be selected.

5. Specify the accuracy required of the metamodel

We specify the range of accuracy required of the output of the metamodel, for its domain of applicability, with respect to both the simulation model and the problem entity. This range depends on the goal of the metamodel and also, perhaps, on the goal of the simulation study. The accuracy required of the metamodel when applied to the simulation model may differ from the accuracy when it is applied to the problem entity. The ranges of accuracies may be specified via the validity measures discussed in the next step.

6. Specify the metamodel validity measures and their required values

We determine the validity measures for the metamodel, along with their required values. (As discussed in §3.7, commonly used validity measures are the absolute error and absolute relative error; their required values are often specified as less than some maximum allowable value.) The validity measures used and/or their required values may be different for the metamodel with respect to the simulation model and with respect to the problem entity.

7. Specify the metamodel

First we select a type of metamodel to be used; examples are polynomial regression models and neural networks. Next, we select a form of the metamodel from the type selected; examples are metamodels that are either first- or second-degree polynomials, and neural networks that have different levels and number of nodes.

8. Review the metamodel specification

The users of the metamodel review the type and form of metamodel specified, to ensure that it is satisfactory for the intended goals of the metamodel.

9. Specify an experimental design

We determine the type of experimental design to be used (e.g., 2^{k-p}) and the design points for which data are to be collected from the simulation model, in order to both fit and validate the metamodel. The resulting I/O data should be partitioned into two parts: one for fitting the

metamodel, and one for validating the metamodel. Determining the experimental design is frequently called a strategic issue in DOE; see the next step.

10. Decide the tactical DOE issues

When the simulation model is random, the number of observations (runs) at each design point must be determined. Further, we may use variance reduction techniques (e.g., common pseudorandom numbers). There may be interaction between Steps 9 and 10; for example, common and antithetic numbers may be selected following the well-known Schruben and Margolin strategy; see Donohue (1995).

11. Review all DOE aspects

We review the decisions made in Steps 9 and 10, to determine whether they are satisfactory for the goal, type, and form of metamodel specified and for the validation of the metamodel.

12. Fit the metamodel

First, we run the simulation model to obtain the I/O data (as specified in Steps 9 and 10) for fitting the metamodel. (We do not yet obtain the data for determining validity; see the next step.) Second, from the data we obtain estimates for the parameter values of the metamodel; for example, least square estimates. Next, we assess (i.e., evaluate) these estimates, using mathematical and statistical criteria. For example, if the metamodel is a linear-regression model, then a statistical analysis can be made to determine if the metamodel is overfitted (i.e., some of the parameters can be set zero). If the assessment is not satisfactory, then we repeat Steps 7 through 12.

13. Determine the validity of the fitted metamodel

First, we run the simulation model to obtain the data (as specified in Steps 9 and 10) for validating the metamodel. Next, we determine if the metamodel satisfies the validity measures with respect to the simulation model (as specified in Step 6), first for the validity data set, and then for the data set used for fitting the metamodel. If the metamodel satisfies the validity measures for both data sets, then the metamodel is considered valid with respect to the simulation model. If the validity measures are not satisfied, then we repeat Steps 7 through 13.

Lastly, we determine the validity of the metamodel with respect to the problem entity. The amount of testing and evaluation depends on the goal of the metamodel. If validity cannot be established with respect to the problem entity, then we repeat Steps 7 through 13.

3. Discussion

In this section we discuss eight specific issues (§3.1 through §3.8) that affect one or more steps of the metamodeling process presented in the previous section.

3.1. Metamodel, Simulation Model, and Problem Entity

It is desirable to determine the validity of a metamodel with respect to the problem entity. We may use the same approaches and techniques used in determining the validity of a simulation model. (These approaches and techniques will not be presented here, since they are readily available in the literature; see §1.) To obtain a high degree of confidence in a metamodel's validity with respect to the problem entity requires that the metamodel and problem entity outputs be compared for numerous experimental conditions over the metamodel's domain of applicability. Unfortunately, this is usually not possible because there is too little data on the problem entity's output: usually the problem entity either is unobservable or it has limited observableness. (If data could readily be obtained from the problem entity, then some other approach -such as experimenting directly on the problem entity or developing a 'metamodel' directly from problem entity data- would most likely be used instead of simulation and metamodeling.) Therefore, the validity of a metamodel is determined by (i) making comparisons between the outputs of the metamodel and the simulation model for numerous experimental conditions, and (ii) between the metamodel and the problem entity, using whatever approaches and techniques are appropriate and feasible. If the problem entity is unobservable, then it is impossible to determine whether the metamodel satisfies any specific numerical accuracy with respect to the problem entity.

We must beware that the difference between the metamodel and the problem entity's responses result from a combination of two approximations: (i) the metamodel is an approximation of the simulation model, and (ii) the simulation model is an approximation of the problem entity (see again Figure 1). These two approximations either add to each other or partially cancel each other. Therefore, when specifying the required metamodel's range of accuracies, these two approximations must be considered. Hence, it is not uncommon to have different specified ranges of accuracy required of the simulation model and of the metamodel.

Whenever the simulation model has continuous inputs and parameters, it is impossible to compare a metamodel and a simulation model over the complete domain of the metamodel's intended domain of application for validity. Instead, DOE is used to determine the data points

(or experimental conditions) to be used in the comparisons discussed in Steps 9, 10, and 13 of the metamodel process (see §2). A metamodel may be valid for one subrange of input values, but not for another. For example, in queueing simulations a first-order polynomial regression may be valid for low traffic rates, whereas higher traffic rates require more sophisticated metamodels; see Cheng and Kleijnen (1997).

A metamodel is modified until it is valid for all experimental conditions tested, using the specified validity measures and their required values. After these comparisons are satisfied, the metamodel is considered valid with respect to the simulation model. Next, the metamodel is validated as appropriate with respect to the problem entity. The appropriateness depends on the goal of the metamodel.

When deciding the validity of any model, either a correct or a wrong decision can be made. A correct decision is made if either a valid model is accepted as being valid or an invalid model is rejected as being valid. A wrong decision is made if a valid model is rejected, called a type I or α error, or if an invalid model is accepted as valid, called a type II or β error. A type II error is the most critical error; therefore, avoiding type II error should be emphasized in validating metamodels. When statistical tests are used to determine validity, sometimes the probabilities of type I and II errors can be calculated. See Balci and Sargent (1981).

An individual factor's importance and significance are related, but different concepts. Significance is a statistical concept. An important factor may be declared non-significant if the variance of the estimated effect is high (because the output has high variance and the total sample size is small): this is another example of type II error. Importance depends on the practical problem that is to be solved. An unimportant factor may be declared significant if the variance of the estimated factor effect is small (in simulation, large sample sizes do occur).

3.2 Four Goals of Metamodeling

Above we distinguished four general goals for metamodels: (i) understanding, (ii) prediction, (iii) optimization, and (iv) aiding in the V & V of a simulation model. We now discuss these goals in more detail.

(i) Understanding

Different degrees of understanding are reflected by different *measurement scales*. There are five scales on which to measure the inputs and outputs of the metamodel, simulation model,

and problem entity:

Nominal; for example, First In First Out (FIFO) versus Shortest Processing Time first (SPT)

Ordinal; for instance, a robot is 'more flexible' than a dedicated machine (this scale is used in statistical rank tests)

Interval; 20° is warmer than 10°, but not twice as warm (arbitrary zero)

Ratio; $2 is twice as much as $1 (absolute zero); but 200 cents is also twice as much as 100 cents

Absolute: 1 server versus 2 servers

For exact definitions of these scales we refer to Kleijnen (1987, pp. 138-141). As more knowledge is acquired, the scale that is actually used becomes more discriminating; for example, in physics the Kelvin scale replaces the Celsius scale.

We distinguish the following *levels of understanding*.

(a) *Directions* of output: does the output increase or decrease for an increase in an input? At least an ordinal scale is then needed. A metamodel used to determine the direction or sign can often be simple; in fact, it may be desirable to develop several simple models instead of one complex model. The signs of the individual parameters should support prior knowledge about the problem entity. For example, in a queueing problem the mean waiting time should increase with traffic rate; so if a first-order polynomial is used and x_1 denotes traffic rate, then its estimated effect $\hat{\beta}_1$ should be non-negative.

(b) S*creening*: which factors have important effects; what is the 'short list' with the most important factors? For this goal, a rather crude metamodel may suffice; of course there is always the danger that such a crude metamodel is misleading. A first-order polynomial, possibly augmented with cross-products (or two-factor interactions) is used by a screening technique called sequential bifurcation; see Bettonvil and Kleijnen (1997). An alternative technique is discussed in Saltelli, Andres, and Homma (1995).

(c) *Main effects, interactions, quadratic effects, and other high-order effects*: what are the effects of a specific factor, possibly in combination with other factors? The type of metamodel may still be simple: first- or second-order polynomials, possibly combined with transformations such as $1/x$ and $\log(x)$. For example, Kleijnen and Standridge (1987) found the bottlenecks in a simulated Flexible Manufacturing System (FMS): the explanation suggested by the metamodel is that among the four machine types simulated, there are two bottleneck machine types that

interact (see $\hat{\beta}_2$, $\hat{\beta}_4$, $\hat{\beta}_{2;4}$ in that reference). Another example is the single-server simulation in Cheng and Kleijnen (1997): a second-degree polynomial can explain the 'exploding' behavior of such a simulation as the traffic rate x approaches unity; even better is the explanation by a first-order polynomial multiplied by the factor $1/(1 - x)$.

We point out that regression analysis aims at more understanding than classic Analysis of Variance (ANOVA) and its extensions, namely Multiple Comparison Procedures (MCPs) and Multiple Ranking Procedures (MRPs). In ANOVA we test whether a given number of system configurations (populations) have the same mean. So the null-hypothesis is: the factor has no effect at all. Regression analysis, however, aims at estimating the magnitudes of the effects and at inter- and extrapolation. MCPs aim at detecting clusters of system configurations with the same mean per cluster. MRPs aim at finding the best system configuration among a limited set of configurations. See Bechhofer, Santner, and Goldsman (1995), Ehrman, Hamburg, and Krieger (1996), and Fu (1994).

Low-order polynomials are also simpler to understand than splines and neural networks. Main effects, two-factor interactions, and quadratic effects are easy to comprehend; they can be easily displayed graphically.

The type of metamodel selected should be appropriate for the type of understanding desired. For example, if it is desired to understand what is occurring inside some system, then a metamodel such as a polynomial over the entire domain of applicability is probably more appropriate than (say) a spline metamodel, which consists of several simple metamodels fitted to subdomains of the domain of applicability. A spline model may be the best choice if the response surface is expected to be highly complex and the understanding desired is what inputs have the most effect on the output.

(ii) Prediction

Prediction requires either the absolute or the relative magnitude of the output. For example, the goals of the simulation study in Kleijnen (1995a) are:

(a) determine whether it is worthwhile to send a ship equipped with sonar into a certain area, for a mine search mission: absolute scale necessary;

(b) determine the relative effects of certain technical sonar parameters: interval or ratio scale needed.

Other examples are the single-server queues with priority rules such as SPT, and queueing

networks in Cheng and Kleijnen (1997). These examples require up to a fourth and sixth-degree polynomial respectively.

In general, if the metamodel is used for prediction, then the type of metamodel must be selected with extreme care. A first reason is that the response surface to be approximated by the metamodel often has complex behavior over the domain of applicability; see Sargent (1991). A second reason is that the accuracy required for a predictive metamodel is usually very high. If, for example, a polynomial is used directly on the simulation data, then it probably will have to be of high order (with numerous interaction terms). However, if a good transformation of data can be found (see Cheng and Kleijnen 1997), then a simpler polynomial regression model can be used. There are several other types of metamodels that can be used instead of polynomial regression models; for example, neural networks and splines; see the references in §1.

To predict an output for a new set of input values, the simulation model itself can be used, if simulation run time is not prohibitive. If, however, simulation time is prohibitive (as may be the case in real-time control), then a metamodel may be used; examples might be found in production scheduling and financial markets.

(iii) Optimization

Optimization is a well-known topic in Operations Research/Management Science. Closely related to optimization is goal seeking: given a target value for the output, find the corresponding input values. There are many mathematical techniques for optimizing the decision variables of nonlinear implicit functions; simulation models are indeed examples of such functions. These functions may include stochastic noise, as is the case in random simulation. Examples of such optimization techniques are sequential simplex search, genetic algorithms, simulated annealing, and tabu search (see Kleijnen 1998). Metamodels for optimization that account for both the mean and the variance of the output is the focus of Taguchi's methods; see Sanchez et al. (1996). This paper, however, concentrates on Response Surface Methodology (RSM); see the four books and the review articles on RSM that are referenced in Khuri (1996b, p. 377).

RSM relies on low-order polynomial regression metamodels. Local marginal effects are estimated to find the direction of improvement. In the local-exploration phase, RSM uses a sequence of first-order polynomial metamodels, combined with steepest ascent search. In the final optimization phase, RSM uses a single second-order polynomial metamodel. Hence, optimization is more demanding than 'understanding' and less demanding than prediction.

(iv) V & V of a simulation model

We present two ways that metamodels can aid in the V & V of a simulation model. The first way is determining whether the metamodel has effects with the *correct signs*: does the output in the metamodel respond to changes in the inputs in the direction that agrees with prior, qualitative knowledge about the simulated problem entity? A simple academic example is a queueing simulation: waiting time increases when the input traffic rate increases. Suppose that for low values of the traffic rate the metamodel is a first-order polynomial. Then the estimated parameter relating these two variables should be positive; if it is not, then there is an error somewhere. We refer to two case studies where the testing of signs lead to detection of errors in computer programs during verification: the ecological simulation in Kleijnen, Rotmans, and Van Ham (1992) and the military simulation in Kleijnen (1995a).

The second way metamodeling may help in the validation of simulation models arises when the problem entity is easily observable. In that case, comparisons can be made between the simulation model's and the problem entity's outputs for numerous experimental conditions over the simulation model's domain of applicability. Since the collection of data on the problem entity is usually costly, it is important to do this data gathering efficiently. So a decision must be made on how many experimental conditions to observe, and where they should be located. This depends on the complexity of the response surface over the domain of applicability. For example, whether the response surface is linear or quadratic guides the number and location of data points needed for comparison; that is, DOE can be applied. The resulting design is a plan for comparing the problem entity to the simulation model for validation of the simulation model. See Chen (1985).

In general, a simulation model is supposed to meet the V & V requirements only for a certain application domain. Hence, V & V of simulation models requires a test plan; we proposed to base that plan on DOE. But DOE is substantially aided by
the use of an explicit metamodel! However, that metamodel should be validated with respect to the simulation model (which is a focus of this paper).

3.3 Classic DOE

We summarize classic DOE. Readers conversant in DOE may skim this subsection. We know, however, that many simulationists do not feel familiar with classic DOE, so we include this

material to make our paper self-sufficient.

Classic DOE assumes a *low-order polynomial* regression metamodel for a single output with normally, identically, and independently (NID) distributed output. For other metamodels no standard solutions are available; instead a computerized search for 'optimal' designs, given the metamodel is performed. For example, Sacks, Welch, Mitchell, and Wynn (1989) assume that the response function is smooth, and that a covariance-stationary process is a valid model for the fitting errors (more specifically, they assume that the closer two factor combinations are in *k*-dimensional space, the more the fitting errors are correlated: exponentially decaying correlation function). They use splines as metamodels. The resulting designs do not show regular geometric patterns. Also see Welch et al. (1992).

So when we apply classic DOE, then the simulation model has *k* factors, and we assume that the metamodel has (i) *k* main effects (say) β_j ($j = 1, ..., k$), (ii) possibly augmented with interactions that range from the $k(k - 1)/2$ two-factor interactions $\beta_{j; j'}$ (with $j < j'$ and $j, j' = 1, ..., k$) to a single interaction among all *k* factors $\beta_{1; 2; ...; j; ...; k}$, and (iii) possibly further augmented with quadratic effects $\beta_{1; 1}$, ..., $\beta_{k; k}$, cubic effects, etc. The total number of regression parameters is denoted by (say) *q*; for example, a first-order polynomial has regression parameters $\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_k)'$ so $q = k + 1$; bold letters denote matrices including vectors. Classic DOE assumes that the number of factor combinations (say) *n* is fixed (sequential DOE makes *n* random; see §3.6). DOE's goal is to select a strategic plan for running experiments with the simulation model. Also see Figure 2, which shows only a single output, *w*, for the simulation model (and *y* for the metamodel).

Mathematically, this plan requires an *n* x *k* *design matrix* $\boldsymbol{D} = (d_{i; j})$ with $i = 1, ..., n$, and $j = 1, ..., k$. For example, consider the first-order polynomial in *x*:

$$y_i = \beta_0 + \beta_1 x_{i; 1} + ... + \beta_k x_{i; k} \; (i = 1, ..., n) \Leftrightarrow y = \boldsymbol{X_1 \beta_1} \qquad (1)$$

where $\boldsymbol{X}_1$ is the *n* x $(k + 1)$ matrix of *independent variables* of the first-order polynomial metamodel, which has row vectors $\boldsymbol{x}_i = (1, x_{i; 1}, ..., x_{i; k})$; we ignore replicated observations (in random simulation, however, there are replications; see §3.4). But the values of $\boldsymbol{X}_1$ are fixed by $\boldsymbol{D}$, since $\boldsymbol{X}_1 = (\mathbf{1}, \boldsymbol{D})$ with $\mathbf{1} = (1, ..., 1)'$ (below we shall see that $\boldsymbol{X}_1$ may also be augmented with cross-products such as $x_j x_{j'}$ and quadratic terms such as x_j^2).

To obtain unique parameter estimates, the design matrix $\boldsymbol{D}$ must meet the following mini-

mum conditions. The number of simulated combinations should not be smaller than the number of metamodel parameters: $n \geq q$. If there are several factors ($k > 1$), then these factors should not be changed simultaneously; otherwise their individual effects cannot be quantified (also see the comment below (2), concerning collinearity).

Classic DOE uses *Ordinary Least Squares* (OLS) fitting. OLS implies use of the L_2 norm, which is a mathematical criterion. For OLS there is much standard mathematical-statistical software. So the OLS estimator (say) $\hat{\beta}$ is computed by *fitting* the specified linear-regression metamodel to its set of I/O data. These data consist of the $n \times q$ matrix of independent regression variables $\boldsymbol{X} = (\boldsymbol{x}_i)$ and the n-dimensional vector of simulation outputs $\boldsymbol{w} = (w_i)$ with $i = 1, ..., n$. Figure 2 shows these data for a single simulation run or factor combination. This gives

$$\min_{\hat{\beta}} \sum_{i=1}^{n} [w_i - y_i(\hat{\beta}, \boldsymbol{X})]^2 \rightarrow \hat{\beta} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{w}. \quad (2)$$

This formula assumes that the inverse exists (non-singular $\boldsymbol{X}'\boldsymbol{X}$): DOE must ensure that $\boldsymbol{X}$ (which is fixed by $\boldsymbol{D}$) is indeed not collinear or ill-conditioned.

In Figure 2, the symbol s_0 denotes the pseudorandom number seed, which plays a role only in *random simulation* models. The I/O of the problem entity is supposed to inspire the selection of the I/O of the simulation model.

Generalizations of OLS are Weighted LS (WLS), and Generalized LS (GLS). WLS uses the standard deviations of the outputs (say) σ_i as weights. In case of variance heterogeneity, the WLS estimator is the 'best linear unbiased estimator' (BLUE); both OLS and WLS give unbiased estimators. GLS is recommended whenever the simulation outputs are correlated, as they are when common pseudorandom numbers are used. For GLS the covariance matrix of the simulation outputs should be estimated. For details on WLS and GLS we refer to Kleijnen (1998). Note that GLS and WLS concern the analysis, not the design of the simulation experiment.

So the classic assumption in case of random simulation is that the simulation responses $\boldsymbol{w}$ have constant variances (say) σ^2, and are independent. Then the covariance matrix of the OLS estimator of the regression parameters is

$$\boldsymbol{\sigma}_{\hat{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\sigma^2. \quad (3)$$

The main diagonal of this matrix consists of the variances of the individual components ($\hat{\beta}_0$, $\hat{\beta}_1$, ...) of $\hat{\boldsymbol{\beta}}$. The goal of DOE may be to minimize these variances through the proper choice of $\boldsymbol{D}$, which fixes $\boldsymbol{X}$, given the form of regression model. It can be proven that an *orthogonal* design matrix is optimal, given the classic assumption. (Other definitions of 'optimal design' are also used; see Kleijnen 1998.)

In *deterministic* simulation the OLS estimator is also the classic estimator. We may interpret this as implying that the *fitting errors* are assumed to have constant variances σ^2, and are independent. (Sacks et al. 1989 assume correlated errors.)

Next we shall discuss four classes of design that are classical in metamodeling: resolution III (or R-3), resolution IV (R-4), resolution-V (R-5), and Central Composite (CC) designs. For certain values of k (number of factors), the design is *saturated*; that is, n (number of factor combinations) is such that it equals q (number of regression parameters). For all four design classes, examples and references are given in Kleijnen (1998).

By definition, a *R-3 design* gives unique estimates (in statistical terms: unbiased estimators) of the $k + 1$ parameters in a first-order polynomial, using only $k + 1$ combinations rounded upward to the next multiple of four: $n = k + 4 - (k \text{ modulo } 4)$. For example, $n = 8$ when $k = 4$, 5, 6, or 7. For $k = 7$ the design is a 2^{7-4} design. For $k = 4$, 5, or 6, columns in $\boldsymbol{D}$ are dropped, namely 3, 2, and 1 column respectively.

A *R-4 design* gives unique estimates of all first-order effects, plus unique estimates of certain sums of two-factor interactions $\beta_{j;j'}$ (more precisely: certain linear combinations of two-factor interactions). For example, when $k = 8$ then a 2^{8-4} design may estimate $\beta_{1;2} + \beta_{4;8} + \beta_{3;7} + \beta_{5;6}$; see Kleijnen (1987, pp. 304-305).

R-4 designs can be simply constructed from R-3 designs. The *foldover principle* means that to the original R-3 design we add the negative or mirror image. So, if the R-3 design has an (n_1 x k) design matrix (say) $\boldsymbol{D}_1$, then the R-4 design has an (n_2 x k) design matrix $\boldsymbol{D}_2$ with $n_2 = 2n_1$ where $\boldsymbol{D}_2$ results from adding $-\boldsymbol{D}_1$ to $\boldsymbol{D}_1$. The foldover principle yields a R-4 design with n equal to $2k$ rounded upward to the next power of two; for example, $n = 16$ when $k = 5$, 6, 7 or 8.

A *R-5 design* gives estimators of all k main effects and all $k(k - 1)/2$ two-factor interactions that are not biased by each other; they may be biased by interactions among three or more factors and by quadratic, cubic, etc. effects. In general, R-5 designs require many factor

combinations. Therefore, in practice, these designs are used only for small values of k.

A *CC design* is meant for a second-degree polynomial. It gives unique estimates of all first-order effects, all two-factor interactions, and all quadratic effects. A CC design is easily constructed, once a R-5 design is available. First, simply add the 'center' or 'base' point. Then add the $2k$ points that result from changing each factor one at a time, once increasing and once decreasing each factor in turn. A disadvantage of CC designs is that its number of combinations is relatively high: $n >> q = 1 + k + k(k - 1)/2 + k$. These designs imply non-orthogonal columns for the $k + 1$ independent variables that correspond with the k quadratic effects and the overall effect β_0.

These four design classes imply a k-dimensional *hypercube* for the experimental domain, expressed in *standardized* factors. So, let the value of the original (non-standardized) factor j in combination i be denoted by $v_{i;j}$. In the simulation experiment $v_{i;j}$ ranges between a lowest value l_j and an upper value u_j; that is, the simulation model is not valid outside that range or in practice that factor can range over that domain only (for example, because of space limitations the number of servers can vary only between one and five). Measure the variation or spread of that factor by the half-range $a_j = (u_j - l_j)/2$, and its location by the mean $b_j = (u_j + l_j)/2$. Now the following standardization is appropriate:

$$d_{i;j} = (v_{i;j} - b_j)/a_j \quad . \tag{4}$$

We recommend standardization indeed; see Bettonvil and Kleijnen (1990).

An alternative to the hypercube is a k-dimensional circle. Moreover certain restrictions may hold; for example, per run the values of the factors add up to 100%. Also see Kleijnen (1998).

If a qualitative factor (nominal scale) has more than two levels ('values'), then several binary variables should be used for coding this factor in regression analysis. For example, the sonar case-study in Kleijnen (1995a) erroneously coded three kinds of bottom (rock, sand, mud) as the values 1, 2, and 3. Most ANOVA software helps avoid such errors.

3.4 Checking the Fit of Metamodels: Classic Analysis

Once having specified a metamodel and having estimated its parameters, it becomes necessary to check possible lack of fit of the metamodel. The measures to be discussed in this subsection, may be applied to any type of metamodel (linear-regression models, neural network, etc.), used to approximate either a random or a deterministic simulation. Random simulation usually has

several observations per factor combination, say m_i. So if N denotes the total number of simulation observations, then N equals $\sum_{i=1}^{n} m_i$. (Recall that we have a single output.)

Notice that in practice, analysts often try to interpret individual effects before they check whether the metamodel as a whole makes sense. However, first the analysts should check if the estimated metamodel is an adequate approximation of the simulation model.

The *fit* of a metamodel to the underlying simulation I/O data may be measured in several ways. The classic measure is the *coefficient of determination* R^2, defined as follows. Let $\mathbf{y}$ denote the metamodel output for the true parameter vector $\boldsymbol{\beta}$, so $\hat{\boldsymbol{y}}$ denotes the metamodel output for the estimated parameter vector. Let $w_{i;\,r}$ denote observation r with $r = 1, ..., m_i$ for factor combination i. Finally, denote the average of all N simulation outputs by $\bar{w}$. Then

$$R^2 = 1 - [\sum_{i=1}^{n} \sum_{r=1}^{m_i} (\hat{y}_i - w_{i,\,r})^2]/[\sum_{1}^{n} \sum_{r}^{m_i} (w_{i,\,r} - \bar{w})^2]. \quad (5)$$

R^2 ranges between zero and one. R^2 equals zero if the metamodel output remains constant for all n factor combinations: $\hat{y}_i = \bar{w}$ with $i = 1, ..., n$. The measure R^2 equals one if all N metamodel outputs equal their corresponding simulation outputs: $\hat{y}_i = w_{i,\,r}$. The latter equality holds if $N = q$ (no degrees of freedom left; saturated design without replications). So, any metamodel with $N = q$ gives *perfect fit*.

The measure R^2 has one important drawback: this measure always increases as more regression variables are added (higher q). Therefore the *adjusted* R^2 is defined:

$$R^2_{adjusted} = 1 - (1 - R^2)(N - 1)/(N - q). \quad (6)$$

Unfortunately, there is no simple lower threshold for this criterion. Also see Kleijnen (1987, p. 193).

A related measure is the *linear correlation coefficient* ρ. Originally this measure was developed to characterize a bivariate normal variable (say) (w, y). In that case, all observations on (w, y) lie on a straight line with intercept α_0 and slope α_1 given by

$$\alpha_0 = E(y) - \alpha_1 E(w); \; \alpha_1 = \rho(w, y)\sqrt{var(y)/var(w)}. \quad (7)$$

Hence, if $\rho(w, y) = 1$, and y and w have equal means and equal variances, then this line has slope one and intercept zero. In metamodeling, however, this measure must be interpreted with care: in Figure 2 the variables w and y are not bivariate normal. For example, the two simulation responses w_1 and w_2 have different means and different variances, which are determined by the vectors $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ (in a queueing simulation w may denote average waiting time, and $\boldsymbol{d}$ may denote the three-dimensional vector of arrival rate, service rate, and queueing discipline). Nevertheless, $\hat{y}$ can be regressed on w. Ideally, simulation and metamodel outputs are equal (see the discussion on R^2). This equality means an intercept of zero and a slope of one. A case study on gas transmission in Indonesia indeed uses such a plot; see Van Groenendaal and Kleijnen (1997).

A different approach to checking the fit of a metamodel is known in the literature as *cross-validation* (we use the word 'validation' in a different sense; 'fitting to subsets' would be a better term for 'cross-validation'). The basic idea is to use the metamodel to predict the outcomes for *new* factor combinations of the simulation model, and to compare these predictions with the corresponding simulation responses. In cross-validation a refinement of this idea is the following.

(i) eliminate one factor combination (say) i with $i = 1, ..., n$ (instead of adding new combinations);

(ii) re-estimate the metamodel from the remaining n - 1 combinations, assuming n - 1 observations suffice (non-saturated design; see §3.3);

(iii) recompute the forecast $\hat{y}_i$ and compare this forecast with the simulation output w_i;

(iv) repeat this elimination for all values of i.

In practice, the relative prediction errors $\hat{y}_i/w_i$ are often used. These errors may be 'eye balled', given the goals of the model. Examples are the deterministic simulation model for a FMS system in Kleijnen and Standridge (1987), and for a coal transport system in Kleijnen (1995c).

So far we have concentrated on cross-validation of the metamodel as a whole. Only if the metamodel as a whole fits well, individual parameters should be examined. So now it becomes interesting to observe how the *estimated individual metamodel's parameters* change, as simulated factor combinations are deleted. Obviously, if the specified metamodel is a good approximation, then these estimates remain stable. Examples are given for polynomial regres-

sion metamodels in the FMS and the coal-transport studies mentioned above. In these metamodels the parameters can be interpreted as factor effects, so the metamodel can be used for explanation, whereas the $\hat{y}_i / w_i$ concern prediction. Other metamodels are harder to interpret.

Diagnostic statistics related to our cross-validation measures are RSTUDENT, PRESS, DEFITS, DFBETAS, and Cook's *D*. These diagnostics are discussed in the general literature on regression analysis, and are computed by modern software such as SAS; see Draper (1994) and Swain (1996).

Eliminating noisy parameters may decrease the variance of the remaining parameter estimators ($\hat{\boldsymbol{\beta}}$) and the corresponding predictions ($\hat{\boldsymbol{y}}$). For example, for an M/M/1 queueing simulation Cheng and Kleijnen (1997) fit a first-order and a second-order polynomial respectively; the former has a variance that is a factor seven smaller than the overfitted latter metamodel! Elimination of unimportant factors may also increase the adjusted R^2. For another example we return to Kleijnen and Standridge (1989): cross-validation of a first-order polynomial with k = 4 gives unstable estimates for the factors 1 and 3 and high values for the relative prediction errors. Therefore the first-order polynomial is replaced by a second-order polynomial for the k = 2 stable factors only; non-stable factor effects are set zero. Now an adequate metamodel results: stable effects ($\hat{\beta}_2$, $\hat{\beta}_4$, $\hat{\beta}_{2;\,4}$) and low prediction errors. In general, the mission of science is to come up with simple explanations: parsimony or Occam's razor.

3.5 Checking the Fit of Linear-Regression Metamodels for Random Simulations

In this subsection we discuss random simulations and linear-regression metamodels (including polynomials). Then the predictor for the simulation output given $\hat{\boldsymbol{\beta}}$ (OLS estimator of regression parameters), and $\boldsymbol{x}_i$ (vector of independent variables in factor combination i) is

$$\hat{y}_i = \boldsymbol{x}_i' \hat{\boldsymbol{\beta}} \quad . \tag{8}$$

Denote the covariance matrix of the estimator $\hat{\boldsymbol{\beta}}$ by $\boldsymbol{\sigma}_{\hat{\beta}}$; see (3). Then the variance of the predictor in the preceding equation is

$$var(\hat{y}_i) = \boldsymbol{x}_i' \boldsymbol{\sigma}_{\hat{\beta}} \boldsymbol{x}_i \quad . \tag{9}$$

If the OLS assumptions do not hold (but the polynomial metamodel does), then GLS should be

applied; see §3.3 and Kleijnen (1998).

Cross-validation implies elimination of I/O combination *i*; we denote this by the subscript -*i*. So the new vector of estimated effects is

$$\hat{\boldsymbol{\beta}}_{-i} = (\boldsymbol{X}_{-i}'\boldsymbol{X}_{-i})^{-1}\boldsymbol{X}_{-i}'\boldsymbol{w}_{-i} \text{ with } i = 1, ..., n . \quad (10)$$

Actually, these *n* recomputations can be avoided, using the original $\hat{\boldsymbol{\beta}}$ and the so-called hat matrix $(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'$; see Kleijnen and Van Groenendaal (1992, pp. 156-157).

Substitution of this last equation (10) into the predictor equation (8) gives the new predictor $\hat{y}_{-i}$. The Studentized cross-validation prediction error is

$$\tilde{t}_{\nu} = \frac{w_i - \hat{y}_{-i}}{[\hat{var}(w_i) + \hat{var}(\hat{y}_{-i})]^{1/2}} \quad (11)$$

where the degrees of freedom ν are unknown. Kleijnen (1992) uses $\nu = m - 1$. In cross-validation this test statistic can be computed for each of the *n* data points (simulation runs, factor combinations). To control the type I error rate, Bonferroni's inequality may be applied.

Note: In deterministic simulation, this last equation gives misleading conclusions. In such simulations the constant variance σ^2 is estimated from the residuals of the linear-regression metamodel. Hence, the worse the metamodel is, the bigger this estimate becomes. But then the denominator in (11) increases (while $\hat{var}(w_i)$ remains zero), so the probability of rejecting this false model decreases!

An alternative to cross-validation is the *F lack-of-fit test*. This test compares the following two estimators of σ^2, assuming a common variance of the simulation outputs (we shall discuss this assumption below). The first estimator is the classic variance estimator based on replication:

$$s_i^2 = \sum_{j=1}^{m_i} (w_{i,j} - \bar{w}_i)^2/(m_i - 1) \quad (12)$$

where the average of the m_i independent simulation replications is $\bar{w}_i$ with $i = 1, ..., n$. Because the true variance is constant, these *n* estimators are averaged or pooled: $\sum_{i=1}^{n} s_i^2/n$. (If m_i is

not constant, then a weighted average is used, with the degrees of freedom m_i - 1 as weights.) Next consider the n estimated residuals, $\hat{e}_i = \bar{w}_i - \hat{y}_i$. These residuals give the second variance estimator, $\sum_{i=1}^{n} \hat{e}_i^2 m/(n - q)$. The latter estimator is unbiased if and only if (iff) the regression model is specified correctly; otherwise this estimator overestimates the true variance. Hence the two estimators are compared statistically through the well-known F-statistic, namely $F_{n-q, N-n}$. The lack of fit is declared significant if this statistic exceeds its upper 1 - α quantile.

Rao (1959) extends this test from OLS to GLS: formulas are given in Kleijnen (1992). The latter reference shows that Rao's test is better than cross-validation if the simulation responses are symmetrically distributed, for example, normally or uniformly distributed. Lognormally distributed responses, however, are better analyzed through cross-validation.

Most linear-regression software also gives the *statistical significance* of an *individual* parameter estimate:

$$t_{N-q} = \hat{\beta}_j / [\hat{var}(\hat{\beta}_j)]^{1/2}, \tag{13}$$

where the numerator follows from (2) and the denominator from (3) and (12). If this statistic is not significant ($|t_{N-q}| < t_{N-q}^{\alpha/2}$), then this regression parameter may be set zero. (Whether it makes sense to compute this significance test for deterministic simulations, may be debated.)

A different type of sensitivity is with respect to *outliers* in the simulation output w in random simulations. The pseudorandom number stream (initialized by s_0) may be atypical, resulting in a value w that has extremely low probability. For example, the event of a sequence of 1,000 consecutive pseudorandom numbers all below 0.01, is possible, but highly unlikely. Therefore the analysts may wish to eliminate this value w when fitting the metamodel. See the general regression literature.

3.6 Checking the Fit of Metamodels: Stagewise DOE

Classic DOE assumes that the n combinations of the design matrix $\boldsymbol{D}$ are executed in one big experiment, that is, not in a sequential or stagewise way. The reason is that DOE was developed in agriculture; in simulation, however, the computer generates the N runs one after the

other (if we ignore parallel computing).

Moreover, classic DOE assumes a polynomial regression metamodel. We suppose that it is uncertain what the correct order (degree) of the polynomial is. Then in general we recommend R-4 designs over R-3 designs. But, the *foldover* principle implies that the analysts may first run a R-3 design (say) $\boldsymbol{D}_1$ with n_1 combinations, to compute the k first-order (or main) effects. Some R-3 designs are saturated, so it is impossible to apply cross-validation. However, certain values of k give $n_1 > q$; for example, $k = 4$, 5 or 6 (so $q = 5$, 6, or 7) requires $n_1 = 8$.

After the R-3 design is run, the mirror design $-\boldsymbol{D}_1$ may be executed. Contrary to classic DOE, the analysts may consider executing the n_1 combinations of $-\boldsymbol{D}_1$ one after another: cross-validation can then be applied. Once all combinations of the foldover design have been run, sums of certain two-factor interactions can be computed. These sums tell *which interactions* explain lack of fit of the first-order metamodel. (In cross-validation the design matrix is no longer orthogonal, so certain optimality properties disappear.)

Suppose (say) factor 1 has an unimportant first-order effect ($\hat{\beta}_1$) and this factor is not involved in an important sum of interactions (no significant $\hat{\beta}_{1;\,2}$, ..., $\hat{\beta}_{1;\,k}$). Then this factor may be removed from the polynomial metamodel, to avoid overfitting. In other words, the R-4 design shows whether a first-order polynomial gives adequate fit of the metamodel: no important (sums of) interactions. But as in any modeling effort, there are no foolproof recipes; for example, factor 1 may still have a quadratic effect $\beta_{1;\,1}$ (or its interaction $\beta_{1,\,2}$ may happen to equal the negative sum of all the other components of the sum that it is part of).

It may be impossible (say, too time-consuming) to run the mirror design $-\boldsymbol{D}_1$. If only one extra simulation run is possible, then we propose to run the center point, $\boldsymbol{d}_{n_1 + 1} = (0, ..., 0)'$, which has k zeroes.

If it turns out that a *second-order polynomial* metamodel is to be fitted, then the R-3 or the R-4 design needs to be expanded to a CC design. Again, stagewise experimentation is possible: add $2k$ axial points plus the center point. The R-4 design may not be a proper subset of the CC design; then some combinations may be used for checking the fit of the second-order polynomial metamodel.

Once the CC design has been run and the second-order polynomial metamodel has been fitted, this model can be checked for lack of fit: third-order effects may be important. Part of the CC design is a fractional two-level factorial (such as a 2^{k-p} design), assuming $k > 4$ (see

Kleijnen, 1987, p. 309). Hence a fraction (for example, a 2^{-p} fraction) of the full 2^k is not yet simulated. As new check-points we recommend these not yet simulated factor combinations. Which ones to select, may be determined randomly, if no other information is available. Note that these extra points require *extra*polation of the fitted metamodel, whereas the center point (used to check a first-order metamodel fitted through a R-3 or a R-4 design) implies *inter*polation. (In general, a metamodel is more reliable when used for interpolation; it may be dangerous when used to extrapolate the simulated behavior far outside the domain simulated in the DOE.)

3.7 Validation Criteria, Measures, and Estimators

By definition, any model (be it a simulation model or a metamodel) is a simplification. Such a simplification must have one or more goals (see §3.2). Even a single goal may be served by different types and subtypes of models. For example, to understand the I/O behavior of a complicated simulation model, a polynomial of either first-order or second-order may be used.

To decide whether to accept a specific model, a *criterion* is necessary. For example, a simple first-order polynomial metamodel is accepted iff the accuracy of this model is adequate. The definition of accuracy depends on the goal of the model. Some case studies (see above) measure the accuracy by the *Absolute Relative Error* or ARE (say) $r_1(w, y) = |(w - y)/w|$ (where w denotes the simulation output and y the metamodel outcome; see again Figure 2). Below we shall sometimes simplify this symbol to r_1 or r. This measure, however, is deficient if w ranges from negative through zero to positive values. Examples are provided by financial analyses that use the Net Present Value (NPV) criterion; see the case-study in Van Groenendaal and Kleijnen (1997).

Therefore another measure may be used, namely the *Absolute Error* or AE (say) $r_2 = |w - y|$ (which is the numerator of r_1). This measure is relevant if, for example, the problem entity is: 'if the target is not detected within five meters, the weapon is ineffective'.

Another popular measure is the *Mean Squared Prediction Error* (MSPE); see Diebold and Mariano (1995). Besides the usual statistical measures , this reference also considers more general ‘economic loss’ functions; for example, asymmetric functions (over- and under-prediction of the same magnitude may have different consequences).

A model is accepted iff the inaccuracy measured by r, remains below some prespecified *threshold* (say) r_{max}, for example, $r_{max} = 0.1$. (Also see §3.1, which speaks of 'a satisfactory range of accuracies'.)

Since a metamodel is a derivative of the underlying simulation model, we may use the same criterion for the metamodel as the one used for the underlying simulation model. So, ideally, the metamodel's output y should be compared with the problem entity's output z; see Figure 2. This gives the criterion $r_1(z, y)$, analogous to $r_1(z, w)$. In practice, however, the problem entity may not yet exist. Even if it does exist, the number of real-life observations on z is usually small, compared with the number of observations on the simulation output w. The latter number is restricted by the computer time required by the simulation model, and the patience of the simulationists and their clients. There are exceptions, however; for example, in the simulation of computer systems the simulation model usually runs slower than the real system, if the latter exists.

Now we focus on comparing the metamodel and the simulation model: $r(w, y)$. The value of this accuracy criterion varies with the 'system configuration', which is determined by the vector of the k factor values $\boldsymbol{d} = (d_1, ..., d_k)'$ (defined by the DOE):

$$r_1(w(\boldsymbol{d}),\ y(\boldsymbol{d})) = |(w(\boldsymbol{d}) - y(\boldsymbol{d}))/w(\boldsymbol{d})|. \tag{14}$$

These different values may be characterized by either the *mean* inaccuracy (say) γ_1 or by the *maximum* inaccuracy (say) γ_2:

$$\gamma_1 = \int_{\boldsymbol{d}} r(w(\boldsymbol{d}),\ y(\boldsymbol{d}))d\boldsymbol{d};\ \gamma_2 = \max_{\boldsymbol{d}}\ r(w(\boldsymbol{d}),\ y(\boldsymbol{d}). \tag{15}$$

The choice between the mean and the maximum again depends on the problem. For example, in the insurance business, an individual policy holder may generate a loss, but on average the insurance company makes a profit. So, if a model is used many times and small errors may compensate large errors, then the mean γ_1 is adequate. Examples of applications of γ_1 are the neural-network metamodels for economic and financial problems in Verkooyen (1996). If, however, a single large error is catastrophic, then the maximum γ_2 is a better characteristic. Examples are provided by nuclear simulation models and econometric time-series models. We shall give more examples below. Note that different criteria or measures for model selection are examined in detail in the statistical monograph by Linhart and Zucchini (1986).

How can the quantities γ_1 and γ_2 be *estimated*? By definition, all possible combinations of the *k* factors in the experimental domain are relevant; see (15). But, the number of combinations goes to infinity, whenever at least one factor is continuous. Therefore a sample of size *n* is observed for the factors: $\boldsymbol{d}_i$ with $i = 1, ..., n$. This is the topic of DOE (see §3.3).

Different *statistics* may be used to estimate the mean, γ_1. Most popular is the sample average, $\bar{r}$. That average has many attractive properties (e.g., it is the maximum likelihood estimator; it has minimum variance), provided certain assumptions hold, such as the NID assumption for *r*. However, when the factors $\boldsymbol{d}$ are selected using DOE, then the values of the factors are fixed (deterministic) so the NID assumption may not hold for the outputs *r*. (Those *n* values of *r* do not have the same mean if one or more factors do affect the output *r*; their histogram is then not a statistical distribution such as $N(\mu_r, \sigma_r)$.) A better statistic may be the sample median (say) $r_{(n/2)}$, which is an order statistic; see Kleijnen (1987). To estimate the maximum γ_2, we propose to take the maximum in the sample of size *n* ; this maximum is denoted by the order statistic $r_{(n)}$. Diebold and Mariano (1995) discuss several statistical tests for comparing predictive accuracy.

We give two applications of the accuracy measure ARE and its maximum, but without an explicit apriori threshold r_{max}. One application is the coal-transportation system-dynamics study in Kleijnen (1995c). The other application is the FMS simulation in Kleijnen and Standridge (1988). These two examples concern deterministic simulation models. (The mean AE is used in a case study on forecasting the dollar/Dutch guilder in Diebold and Mariano (1995); that case study, however, concerns an econometric time-series model.)

The literature on *random* simulation models focusses on the precision of a single characteristic (e.g., the mean) of a single population of simulation outputs, not on the *n* means of the *n* populations defined for a specific simulation output (e.g., the steady-state waiting time). The latter case holds for metamodels, whereas the former case holds for precision measured by the confidence interval half-width. This width is expressed in either absolute units (say, seconds in waiting time studies) or a percentage (relative precision). When using metamodels in random simulation, we emphasize that these simulation models have intrinsic noise: while the input vector $\boldsymbol{d}$ is kept constant, the simulation output *w* still shows variation. This variation may be measured by the variance (say) $var(w \mid \boldsymbol{d}_i)$ with $i = 1, ..., n$, estimated through s_i^2 defined in (12). Note that this variance estimator is optimal if the NID assumption holds; this assumption

also gives the classic confidence interval for the population mean, based on the Student statistic, t_v.

Consequently, in random simulation it is no longer obvious what *the* simulation output *w* is, when measuring the accuracy; see (14). Actually this output has a statistical distribution that is determined by the simulation model. This distribution may be characterized through different quantities, such as its mean E(*w*) and its 90% quantile, denoted by $w_{(.90)}$. These quantities may be estimated based on simulation replications (alternatives are renewal or regenerative analysis, spectral analysis, standardized time series; see Alexopoulos and Seila 1998). Then in (14) *w* becomes the estimated mean (say) $\bar{w}$ or the estimated quantile or order statistic $w_{(.9m)}$ where *m* denotes the number of replications for the factor combination.

In random simulation we may formulate a null-hypothesis H_0 and an alternative hypothesis H_1 (analogous to r_2 and r_{max}):

$$H_0: |E(w) - E(y)| \leq \delta; \; H_1: |E(w) - E(y)| > \delta \qquad (16)$$

where the value of the threshold δ depends on the problem. Usually such a hypothesis is tested through a Student statistic, t_v. This statistic resembles the ARE, but in random simulation the standardization is done by the standard error $[\hat{var}(\hat{y} - \bar{w})]^{1/2}$, not by the simulation output itself, *w*. Actually, several hypotheses need to be formulated, namely, a hypothesis such as (16) for each of the *n* combinations of factor values. Their simultaneous testing may use Bonferroni's inequality. Also see Diebold and Mariano (1995).

3.8. Validation of Metamodels

The amount of effort devoted to the validation of a metamodel depends on the goal of the metamodel. If that goal is 'understanding', then a 'reasonable' (medium) validation effort should be spent. However, when the goal is 'prediction', then extensive validation of the metamodel should be performed. When the purpose is 'optimization', then a series of metamodels are usually developed; testing for validity may then be limited to the last few metamodels, which should be tested extensively. Finally, if the goal is 'aiding in V & V of a simulation model', then validation is usually conducted only with respect to the simulation model; there is generally no

reason to test the validity of the metamodel with respect to the problem entity.

So the validation of a metamodel is first performed with respect to the simulation model. If the simulation model is deterministic, then either the mean inaccuracy γ_1 or the maximum inaccuracy γ_2 is used as a criterion (see §3.7). For random simulation models, either the Studentized statistic $\tilde{t}_v$ defined in (11) combined with Bonferroni's inequality or the lack-of-fit F-statistic is used (see§3.5). This validation should be conducted for all goals; however, the amount of testing for validity will vary depending on the goal.

The data used for validating a metamodel with respect to the simulation model must first be generated. In steps 9 and 10 of our methodology, we developed a DOE that provided for data generation in two parts: one part for fitting a metamodel, and one part for validating the fitted metamodel. Prior to generating the data for validation, the DOE needs to be reviewed. It may need to be modified because the original DOE was 'expanded' in fitting and checking the fit of the metamodel. The DOE should allow testing during the validation for a higher-order model than the fitted metamodel. Note that we use a separate set of data for validating the meta-model, and that we use sequential experimentation. After the metamodel has been tested and it has passed validity with the new generated data, the data used for fitting and testing the fitted model may also be used for additional validity testing. This is especially worthwhile if the criteria measures used in validity are different than the criteria used for fitting the metamodel.

After a metamodel is validated with respect to the simulation model, the validation of the metamodel with respect to the *problem entity* is conducted. The extend of this validation again depends on the goal of the metamodel. The same methods and techniques used to validate simulation models (references were given in §1) can be used to validate metamodels with respect to the problem entity. The γ_1 and the γ_2, and the statistical formulas given above for criteria testing, can now be used for testing for validity; however, the symbol *w* (simulation output) is replaced by *z* (problem entity output). If the problem entity is unobservable or allows only limited data collection, then it is usually impossible to obtain a high amount of confidence in the metamodel's validity. (This is analogous to what occurs in the validation of a simulation model.). Suppose, however, that the problem entity is observable and data have been collected on it and used for determining the validity of the simulation model (which we have assumed to have been validated in this paper). Then it seems reasonable to use these same data when testing the validity of the metamodel with respect to the problem entity. The resulting statisti-cal confidence level seems to deserve more research.

4. A Procedure for Linear-regression Metamodeling

Most simulation models have *multiple outputs*; for example, customer's queueing time and server's idle time, or mean and 90% quantile of waiting time. In practice, multiple outputs are handled through the application of the techniques of this paper *per* output type. Khuri (1996b, p. 385) proves that the BLUE of the factor effects for the various responses remains the same as the OLS estimators obtained from fitting the linear-regression metamodels per individual output, assuming a single experimental design is used for the data generation of all outputs. Ideally, the design should account for the presence of multiple outputs; see Khuri (1996a, 1996b) and Kleijnen (1987). Simultaneous inference may be taken care of through Bonferroni's inequality. Optimization in the presence of multiple responses is discussed in Kleijnen (1998). Optimization accounting for both the mean and the variance of the output is the focus of Taguchi's methods; see Sanchez et al. (1996). Note that the term 'multiple regression analysis' refers -not to the number of outputs- but to the presence of multiple independent variables $\boldsymbol{x}$.

The analysts should use DOE when doing the strategic planning of the simulation experiment. Classic DOE assumes a given order of the polynomial metamodel, but the analysts should plan for possible changes in that metamodel, as the fitting and validation progress. For example, two-factor interactions may need to be added to the initial first-order polynomial metamodel, and transformations (such as the logarithmic one) may be tried.

An expedient strategy may start assuming a first-order polynomial metamodel for the k factors. This model requires a minimum number of factor combinations, namely $n \approx k + 1$. Which combinations to simulate may be fixed by a R-3 design. But planning ahead means accounting for the possible need to add two-factor interactions. This need can be detected during the fitting process, using cross-validation and the like. If this need does arise, the R-3 design can be easily augmented to a R-4 design, using the foldover principle.

If the R-3 design is saturated, then cross-validation requires that one or more extra combinations be added. We recommend selecting these extra combinations from the corresponding R-4 design.

If the metamodel does not pass the fitting and validation tests, then -instead of adding interactions and simulating the combinations of the augmented design- we may try transformations (such as $\ln(x)$) or disaggregate an independent variable into its components (for example, disaggregate traffic rate into arrival rate and service rate).

In Figure 3 we give a flowchart for this metamodeling process. The figure primarily covers Steps 9 through 13 of the methodology given in §2, but now restricted to linear-regression metamodels. After validating the metamodel with respect to the simulation model, this metamodel should be validated with respect to the problem entity; to save space, these steps are not displayed in the figure.

INSERT Figure 3: A procedure for linear-regression metamodeling

5. Research Issues

We defined *fitting* as the process of applying mathematical/statistical procedures and measures to the I/O data of the metamodel; *validation* was defined such that information on the underlying problem entity -simulated or real- be used. Consequently, validation requires domain knowledge. Its measures may be mean inaccuracy γ_1 or maximum inaccuracy γ_2; see (15). To the best of our knowledge, no research has yet been done, on how to compute the metamodel's parameters such that γ_1 or γ_2 is minimized.

A few academic studies use other mathematical norms than L_2 (see Kleijnen 1987). Well-known alternative norms are the L_1 (absolute fitting errors) and the L_∞ (Tchebyshev) norms:

$$L_1\text{: } \min_{\hat{\beta}} \sum_{i=1}^{n} |w_i - y_i(\hat{\beta})|; \; L_\infty\text{: } \min_{\hat{\beta}} \max_i |w_i - y_i(\hat{\beta})|. \tag{17}$$

Current practice is as follows. Simulationists use a convenient mathematical norm to *fit* a specific metamodel to the simulation I/O data. Next, they often use a *different norm* when they examine the resulting fit and validity of the metamodel! For example, they may apply L_2 when fitting the metamodel, but L_∞ when determining whether this model is adequate. Examples are provided by the case studies mentioned above: Kleijnen (1995a) and Kleijnen and Standridge (1988) use OLS to compute the estimated parameter vector $\hat{\beta}$, but they use the Tchebyshev norm to decide on the adequacy of the resulting metamodel. Verkooyen (1996), however, uses the same norm, namely L_2 in both fitting and validating his neural network metamodel. To the best of our knowledge no research has yet been done, on how to compute the estimated parameters of the selected type of metamodel such that γ_1 or γ_2 is minimized.

In random simulations, cross-validation and the lack-of-fit F-statistic are usually applied to test *zero* lack of fit. However, we may wish to test for a non-zero threshold ($\delta > 0$). The question is whether a non-central F-statistic is adequate.

We discussed inaccuracies of the metamodel relative to the simulation model, and relative to the problem entity. The latter type of error has not been investigated in the literature. Further research seems necessary.

Different goals of metamodeling (understanding, prediction, optimization, V & V of simulation) require different metamodel types and different accuracies. More research on this topic is needed.

Multi-variate regression analysis and multi-variate DOE for metamodeling deserve more research. Metamodels that account for both the mean and the variance of the output have already been the focus of Taguchi's methods; more research is needed.

A procedure such as outlined in Figure 3 for linear-regression metamodels should also be developed for other metamodel types such as neural networks and splines.

6. Summary

We explored the relationships among metamodels, simulation models, and problem entities, while distinguishing between fitting and validating of metamodels. We covered several metamodel types, including linear-regression and neural networks, but we focussed on polynomial metamodels.

In general, validation of a model requires that the goals of that model be kept in mind. For a metamodel we distinguished four types of goals: (i) understanding, (ii) prediction, (iii) optimization, and (iv) V & V of the underlying simulation model.

We proposed a methodology resulting in a process with thirteen steps. This process includes classic DOE and standard measures of fit, such as the R-square coefficient and various cross-validation measures. The process, however, also includes stagewise DOE and several validation criteria, measures, and estimators.

References

Alexopoulos, C. and A. Seila (1998), Output data analysis. *Handbook of Simulation*, edited by J.

Banks, Wiley, New York

Balci, O. and R.G. Sargent (1981), A methodology for cost-risk analysis in the statistical validation of simulation models. *Communications of the ACM*, 24,4, pp. 190-197

Barton, R.R. (1993), New tools for simulation metamodels. IMSE Working Paper 93-110, Department of Industrial and Management Systems Engineering, Penn State University, University Park, PA 16802

Barton, R.R. (1994), Metamodeling: a state of the art review. *Proceedings of the 1994 Winter Simulation Conference*, eds. J.D. Tew, S. Manivannan, D.A. Sadowski, and A.F. Seila, pp. 237-244

Bechhofer, R., T. Santner, and D. Goldsman (1995), *Design and analysis of experiments for statistical selection, screening, and multiple comparisons*. John Wiley, New York

Bettonvil, B. and J.P.C. Kleijnen (1997), Searching for important factors in simulation models with many factors: sequential bifurcation. *European Journal of Operational Research*, 96, no. 1, January 1997, pp. 180-194

Chen, B. (1985), *A statistical validation procedure for discrete event simulation over experimental regions*. Ph.D. Dissertation, Department of Industrial Engineering and Operations Research, Syracuse University, Syracuse, NY 13244

Cheng, R.C.H. and J.P.C. Kleijnen (1997), Improved designs of queueing simulation experiments with highly heteroscedastic responses. Forthcoming in *Operations Research*

Diebold, F.X. and R.S. Mariano (1995), Comparing predictive accuracy. *Journal of Business & Economic Statistics*, 13, no. 3, pp. 253-263

Donohue, J.M. (1995), The use of variance reduction techniques in the estimation of simulation metamodels. *Proceedings of the 1995 Winter Simulation Conference*, edited by C. Alexopoulos, K. Kang, W.R. Lilegdon, and D. Goldsman, pp. 195-199

Draper, N.R. (1994), Applied regression analysis; bibliography update 1992-93. *Communications in statistics, theory and methods*, 23, no. 9, pp. 2701-2731

Ehrman, C.M., M. Hamburg, and A.M. Krieger (1996), A method for selecting a subset of alternatives for future decision making. *European Journal of Operational Research*, 96, pp. 407-416

Friedman, L.W. (1996), *The simulation metamodel*. Kluwer, Dordrecht, Netherlands

Fu, M.C. (1994), Optimization via simulation: a review. *Annals of Operations Research*, 53, pp. 199-247

Huber, K-P., M. R. Berthold, and H. Szczerbicka (1996), Analysis of simulation models with fuzzy graph based metamodeling. *Performance Evaluation*, 27 & 28, pp. 473-490

Khuri, A.I. (1996a), Analysis of multiresponse experiments: a review. *Statistical design and analysis of industrial experiments*, edited by S. Ghosh, Marcel Dekker, New York, pp. 231-246

Khuri, A.I. (1996b), Multiresponse surface methodology. In: *Handbook of Statistics, vol. 13*, edited by S. Ghosh and C.R. Rao, Elsevier, Amsterdam, pp. 377-406

Kleijnen, J.P.C. (1998), Experimental design for sensitivity analysis, optimization, and validation of simulation models. *Handbook of Simulation*, edited by J. Banks, Wiley, New York. (Preprint: CentER Discussion Paper, no. 9752.)

Kleijnen, J.P.C. (1995a), Case study: statistical validation of simulation models. *European Journal of Operational Research*, 87, no. 1, pp. 21-34

Kleijnen, J.P.C. (1995b), Verification and validation of simulation models. *European Journal of Operational Research*, 82, no. 1, pp. 145-162

Kleijnen, J.P.C. (1995c), Sensitivity analysis and optimization of system dynamics models: regression analysis and statistical design of experiments. *System Dynamics Review*, 11, no. 4, pp. 275-288

Kleijnen, J.P.C. (1992), Regression metamodels for simulation with common random numbers: comparison of validation tests and confidence intervals. *Management Science*, 38, no. 8, pp. 1164-1185

Kleijnen, J.P.C. (1987), *Statistical tools for simulation practitioners*. Marcel Dekker, Inc., New York

Kleijnen, J.P.C. and W.J.H van Groenendaal (1992), *Simulation: a statistical perspective*. John Wiley, Chichester (England)

Kleijnen, J.P.C., G. van Ham and J. Rotmans (1992),Techniques for sensitivity analysis of simulation models: a case study of the CO2 greenhouse effect. Simulation, 58, no. 6, pp. 410-417

Kleijnen, J.P.C. and C. Standridge (1988), Experimental design and regression analysis: an FMS case study. *European Journal of Operational Research*, 33, no. 3, pp. 257-261

Linhart, H. and W. Zucchini (1986), *Model selection*. Wiley, New York

Pierreval, H. (1996), A metamodel approach based on neural networks. *International Journal in Computer Simulation*, 6, no.3, pp. 365-378

Rao, C.R. (1959), Some problems involving linear hypothesis in multivariate analysis. *Biometrika*,

46, pp. 49-58

Sacks, J., W.J. Welch, T.J. Mitchell and H.P. Wynn (1989), Design and analysis of computer experiments (includes comments and rejoinder). *Statistical Science*, 4, no. 4, pp. 409-435

Saltelli, A,, T.H. Andres, and T. Homma (1995), Sensitivity analysis of model output; performance of the iterated fractional factorial design method. *Computational Statistics & Data Analysis*, 20, pp.387-407

Sanchez, S.M., P.J. Sanchez, J.S. Ramberg, and F. Moeeni (1996), Effective engineering design through simulation. *International Transactions Operational Research* , 3, no. 2, pp. 169-185

Sargent, R.G. (1996), Verifying and validating simulation models. *Proceedings of the 1996 Winter Simulation Conference*, eds. J.M. Charnes, D.M. Morrice, D.T. Brunner, and J.J. Swain, pp. 55-64

Sargent, R.G. (1991), Research issues in metamodeling. *Proceedings of the 1991 Winter Simulation Conference*, eds. B.L. Nelson, W.D. Kelton, and G.M. Clark, pp. 888-893

Schlesinger, S. et al. (1979), Terminology for Model Credibility, *Simulation*, 32, 3, pp. 103-104

Swain, J.J. (1996), Number crunching: 1996 statistics survey. *OR/MS Today*, 23, no. 1, pp. 42-55

Van Groenendaal, W.J.H. and J.P.C. Kleijnen (1997), On the assessment of economical risk: factorial design versus Monte Carlo methods. *Journal of Reliability Engineering and Systems Safety*, 57, no. 1, 1997, pp. 103-105

Verkooyen, W.J.H. (1996), *Neural networks in economic modelling*. Ph.D. thesis, CentER, Tilburg University, Tilburg

Welch, W. J., R. J. Buck, J. Sacks, H. P. Wynn c.s. (1992), Screening, predicting, and computer experiments. *Technometrics*, Vol. 34, No. 1, pp. 15-25

Yu, B. and K. Popplewell (1994), Metamodel in manufacturing: a review. *International Journal of Production Research*, 32, no. 4, 1994, pp. 787-796

Zeigler, B. (1976) *Theory of modelling and simulation.* New York: Wiley Interscience

Figure 1: Metamodel, simulation model, and problem entity

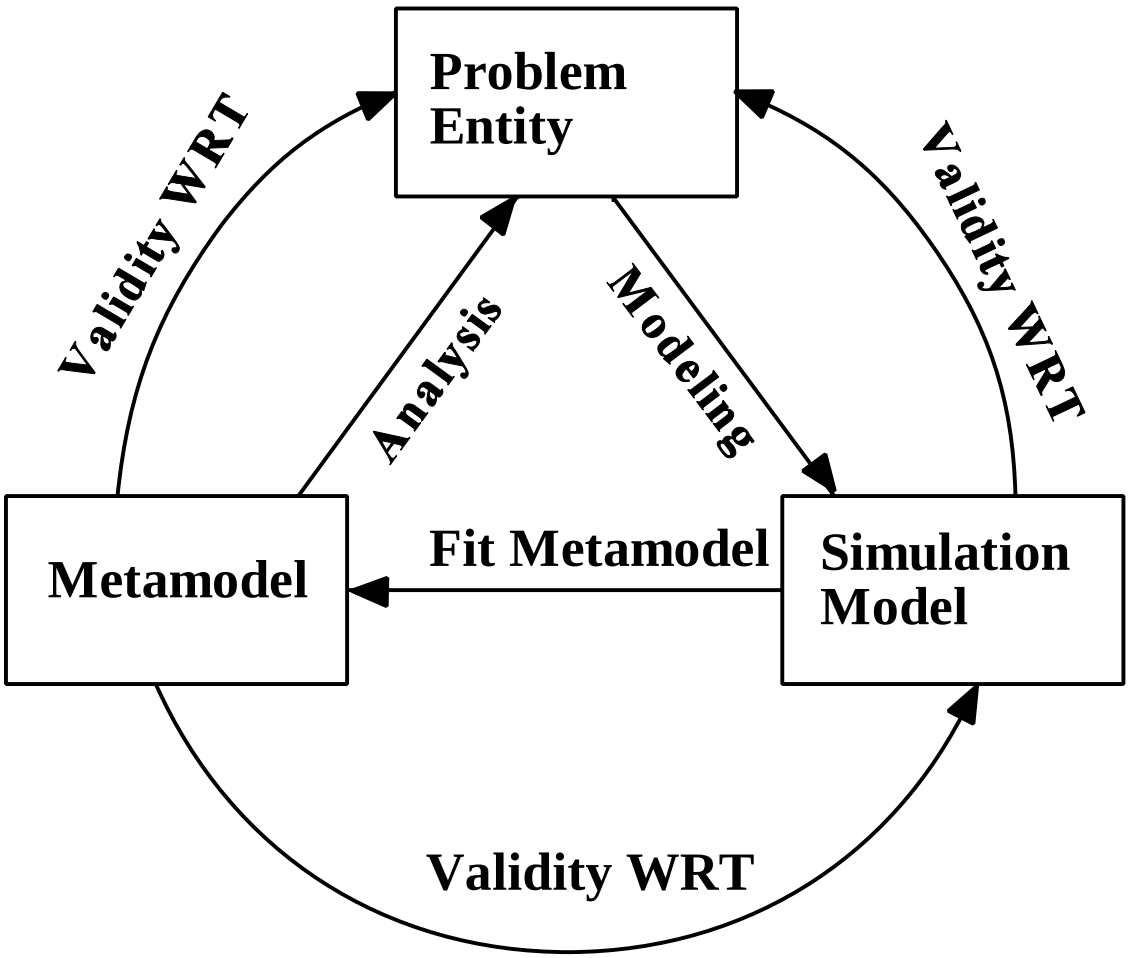

Figure 2: I/O data of problem entity, simulation model, and metamodel

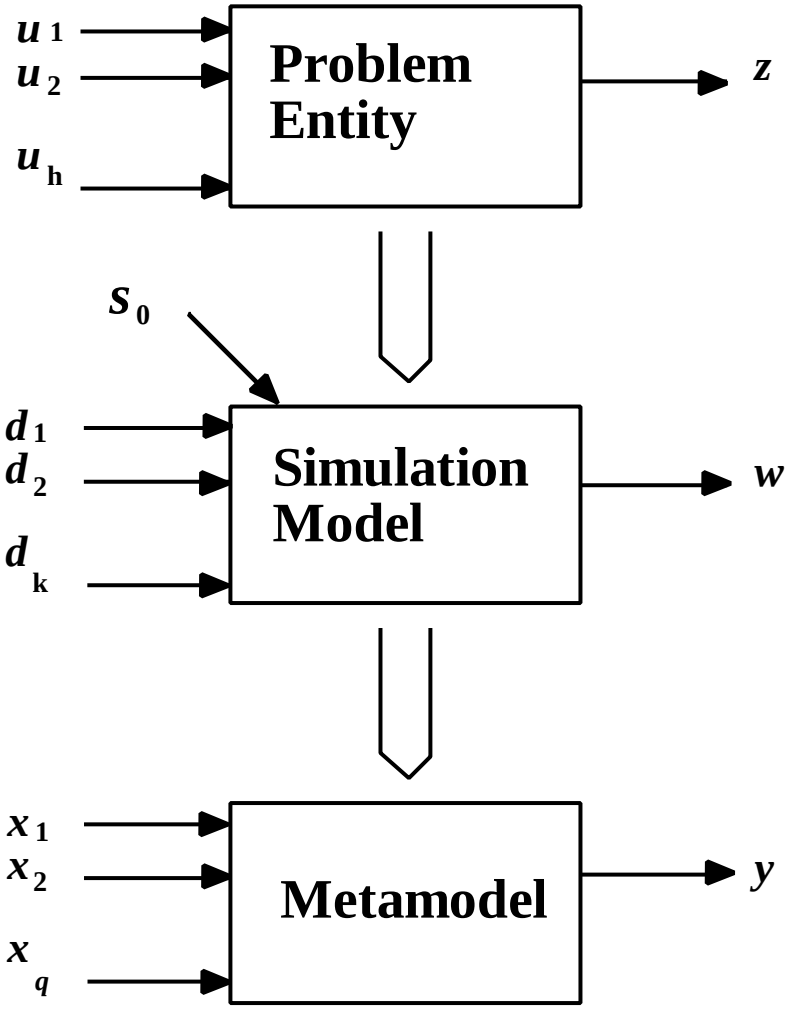

Figure 3: A procedure for linear-regression metamodeling